(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
13 December 2001 (13.12.2001)

PCT

(10) International Publication Number
WO 01/95101 A2

(51) International Patent Classification[7]: G06F 9/38

(21) International Application Number: PCT/US01/10839

(22) International Filing Date: 30 May 2001 (30.05.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/586,190 2 June 2000 (02.06.2000) US

(71) Applicant: SUN MICROSYSTEMS, INC. [US/US]; 901 San Antonio Road, Palo Alto, CA 94303 (US).

(72) Inventors: TREMBLAY, Marc; 140 Hanna Way, Menlo Park, CA 94025 (US). YELURI, Sharada; 1720 Fumia Drive, San Jose, CA 95131 (US). CHAN, Jeffrey, Meng, Wah; 1984 Latham Street #10, Mountain View, CA 94040 (US).

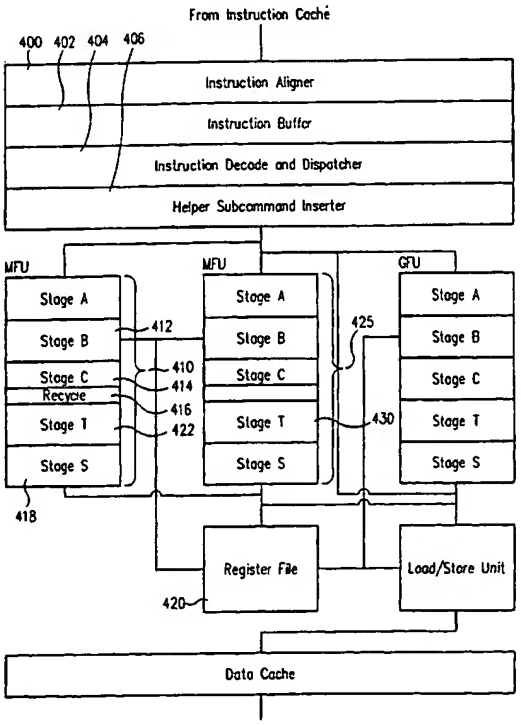
(74) Agent: MCKAY, Philip, J.; Gunnison, McKay & Hodgson, L.L.P., Suite 220, 1900 Garden Road, Monterey, CA 93940 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European

[Continued on next page]

(54) Title: SYNCHRONIZING PARTIALLY PIPELINED INSTRUCTIONS IN VLIW PROCESSORS



(57) Abstract: AVLIW processor has multiple pipelines (410, 425) for execution of subcommands of VLIW instructions in parallel. Each pipeline has at least one execution stage (412, 414) and a trap stage (422, 430). At least one can operate on operands of a first and a second word length, the second word length longer than the first, the first word length is the same as a data path width of the pipeline (410, 425). Execution of operations on the operands of the second word length requires multiple cycles in at least one execution stage (412, 414) of the pipeline. An instruction decoder (404) decodes subcommands of a sequence of VLIW instructions into pipeline subcommands, and dispatches these to the first and second pipelines (410, 425), the instruction decoder (404) injects at least one helper subcommand into the first pipeline (410) when a first subcommand of the VLIW instruction operates on operands of the second word length. The instruction decoder also inserts no-operation helper subcommands into the second pipeline (425) when necessary to ensure that information associated with the first subcommand enters a trap stage (422) of the first pipeline (410) synchronously with information associated with a second subcommand of the same VLIW instruction and dispatched to the second pipeline (425) reaching a trap stage (430) of the second pipeline (425). These no-operation helper subcommands maintain synchronous arrival of information at the trap stages (422, 425) even if the first subcommand operates on operands of the second word length and the second subcommands operates on operands of the first word length.

WO 01/95101 A2

patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii)) for the following designations AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG)*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii)) for the following designations AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG)*

Published:

— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

Synchronizing Partially Pipelined Instructions in VLIW Processors

Field of the Invention

The invention relates to the field of very-long instruction word (VLIW) processor architecture. In particular, the invention relates to synchronization of subcommands in the pipelines of VLIW machines.

Background of the Invention

Most machines, including all high-performance machines, manufactured today are pipelined to at least some extent. A pipeline is generally a hardware execution unit that provides multiple stages of execution, each stage of execution occupying one or more clock cycles. Further, there may be several instructions, each at a different stage of the multiple stages of execution, executing simultaneously.

Modern computing machines often possess two or more execution unit pipelines. Each of these pipelines provides multiple stages of execution. For example, a processor may have an integer execution pipeline, for executing integer subcommands and a floating point execution pipeline, for executing floating point subcommands. Often, it is possible for an integer execution pipeline and a floating-point execution pipeline of a machine to be executing one or more stages of subcommands simultaneously.

A late stage often found in typical execution pipelines is a "trap" stage. The trap stage is where it is determined whether a processor exception (including any of the trap conditions in the IEEE 754 specification) is to occur, or an interrupt is to suspend execution of a currently executing instruction sequence.

When a trap or interrupt, occurs, it is often necessary to determine the state of the processor at

the time the trap or interrupt occurred. It is preferable that traps be handled precisely, such that they can be easily diagnosed, possible corrections made, and execution resumed. If traps are to be handled precisely, it is desirable that no prior-state data be overwritten by any current or later instruction before all pipelines executing the current instruction reach the trap stage, when it can be determined if a trap is required.

It is believed by many engineers that a processor having a large number of pipelines capable of execution in parallel can provide better overall performance than one that has fewer pipelines - provided that instructions can be decoded, operands fetched, and these fed to the pipelines in parallel and at a high rate.

Unfortunately, many processors digest binary instruction languages that have no inherent parallelism. For processors that execute these binary languages to execute instructions in parallel, they must parse their instruction sequence and discover which instructions can be executed in parallel, a non-trivial task. Further, this parsing for potential parallelism is done at execution time, and therefore must be done quickly by very complex hardware.

VLIW processors normally execute a binary instruction language that has explicit parallelism, where each instruction may incorporate subcommands for simultaneous execution in parallel in separate pipelines. This generally requires more bits per binary instruction word than required on conventional processors because each subcommand requires a bit field in the instruction word, these instruction words therefore become very long, hence the term Very Long Instruction Word.

VLIW processors allow separation of the parsing for potential parallelism from execution; this parsing

may therefore occur in a separate instruction translation unit, or may be done at compile time. Parsing at compile time for potential parallelism has the advantage that it can permit use of simpler hardware than otherwise required for the same high performance.

Many processors have pipelines that do not execute all instructions in exactly the same number of clock cycles. It is known that division generally requires more clock cycles than does multiplication. Further, it is known that integer operations are much simpler to execute than are floating point operation, hence floating point pipelines generally require more clock cycles to perform an addition than do integer pipelines. The number of clock cycles a pipeline takes to execute an instruction is the latency of the pipeline. While extra stages may be added to integer pipelines so that they have the same latency as a floating point pipeline of the same machine, and extra stages may be added to a floating point pipeline so that all instructions have the same latency, this is known to be inefficient. It is desirable that subcommands executing in a pipeline complete as early as possible unless a dependency requires that they wait for another pipeline to complete execution.

Many modern processors, including VLIW processors, are optimized for execution of thirty-two-bit data.

It is also known that sixty-four-bit-operand instructions, such as double-precision floating point instructions as described in the IEEE-754 floating point specification, can be executed in hardware that greatly resembles thirty-two-bit execution hardware; however more clock cycles are typically required for sixty-four-bit data than are required for thirty-two-bit data. For example, a sixty-four-bit multiply may be executed in a thirty-two-bit array multiplier stage

of an integer pipeline by making four passes through the array multiplier, while a thirty-two-bit multiply requires only one pass through the array multiplier.

It is known that execution of a sixty-four-bit instruction in a pipeline may be controlled by passing a base instruction into the pipeline, followed by a helper instruction. A base instruction may, for example, process the low half of a sixty-four-bit addition, saving the carry output from the addition. A following helper instruction for the pipeline may then process the high half of the sixty-four-bit addition, injecting the saved carry during the addition.

VLIW processors provide for parallel execution of subcommands of instructions in multiple, often dissimilar, pipelines. These subcommands tend to complete at slightly different times, especially if some are thirty-two-bit and some are sixty-four-bit and the pipelines are optimized for thirty-two-bit data. Worse, a given pipeline may complete some subcommands substantially more quickly than others. Hence, a given pipeline may have a latency that varies from instruction to instruction.

In order to handle traps precisely, it is desirable that all subcommands of a particular VLIW instruction reach the trap stage of the pipelines simultaneously. In order to maintain efficient high-speed operation, it is desirable that no subcommand wait any longer than the actual latency of any other subcommands of the same instruction. It is therefore desirable to stall, or introduce a controlled, minimum, delay, into those pipelines that complete their subcommands of an instruction first such that they reach the trap stage simultaneously with those pipelines that complete their subcommands of the instruction last.

Summary of the Invention

Hardware for stalling those pipelines of a VLIW processor that complete subcommands of an instruction first such that those faster pipelines reach the trap stage simultaneously with those slower pipelines that complete their subcommands of the instruction last is described. This hardware relies on a decode of the instruction to generate stall helper instructions, which are then injected into those pipeline stages requiring stalls, such that the valid results in each pipeline reaches the trap stage simultaneously with those pipelines that execute their subcommands of the instruction more slowly.

The foregoing and other features, utilities and advantages of the invention will be apparent from the following more particular description of a preferred embodiment of the invention as illustrated in the accompanying drawings.

Brief Description of the Drawings

Figure 1 is a block diagram of a computer system having a VLIW processor;

Figure 2, a block diagram of a prior-art VLIW processor having several pipelines that does not enforce precise traps;

Figure 3, a timing diagram of data flow in the pipelines of a VLIW processor, showing how subcommands tend to reach the trap stage at different times if no stalls are injected;

Figure 4, a block diagram of a VLIW processor incorporating the present invention;

Figure 5, a timing diagram of data flow in the pipelines of the preferred embodiment of a VLIW processor of the present invention, showing the stall states required to align the trap stages of several pipelines after the execution of the instruction; and

Figure 6, a timing diagram of data flow in the pipelines of a VLIW processor of the present invention, showing the stall states required to align the trap stages of several pipelines during execution of the instruction.

Detailed Description of the Preferred Embodiment

A computer system has at least one processor 100 (Figure 1) having internal first level cache. The system also has a second level cache 101 and may, but need not, have a third level cache 102. There may, but need not, be an additional processor 105, having its own second level and optional third level caches (not shown). References by the processor 100 that are not satisfied from cache are directed over a high speed local bus 106 to a main memory 107 or through a bus bridge 108 to a system bus 109, which is preferably a PCI bus.

Attached to the PCI bus is a storage controller 115, typically of the Ultra Wide SCSI type, for connection to one or more storage subsystems 116. The storage subsystems 116 typically include a CD reader and/or writer and a disk drive; multiple disk drives may be utilized, as may other peripherals, like RAID storage systems and tape drives. Many computer systems also have a video display subsystem 118, a network interface 120, a USB (universal serial bus) interface 122, as well as keyboard, mouse, serial, printer, and floppy disk ports 124.

The first level cache of the processor 100 may be implemented as separate instruction cache 126 and data cache 128; alternatively these may be combined into a single fast combined cache.

The processor 100 of the computer system may be a VLIW processor. In a VLIW processor, instructions from the instruction cache 126 are aligned by an instruction aligner 200 (Figure 2), and buffered in an

instruction buffer 202. Instructions are then processed by an instruction decoder and dispatcher 204 and dispatched to the various execution pipelines 206, 208, and 210 of the processor. For clarity, figure 2 illustrates three pipelines of a machine that may have more than three pipelines. The pipelines illustrated have operand fetch stages 212, 214, and 216 that are connected to and may fetch operands from a register file 218, and operand store stages 220, 222, and 224 that are connected to and may store results to the register file 218. The processor also has a load/store unit 226 for transferring between the data cache 128 and register file 218.

Let us assume that a VLIW instruction dispatches a thirty-two-bit subtract subcommand to one pipeline, and sixty-four-bit add subcommand to another pipeline. Further, assume, as in the preferred embodiment of the present invention, that sixty-four-bit operations are performed through execution of a sequence of thirty-two-bit sub-operations. As illustrated in Figure 3, the sixty-four-bit subcommand then requires a pair of fetch operations 300 to fetch the operands, each taking a cycle in the fetch stage of the pipeline upon which it executes, and a pair of operate cycles 301, while the thirty-two-bit subcommand requires fewer cycles 302 to fetch operands and operate 303 upon the operands because the thirty-two-bit operands match the width of the datapath of the pipeline. It is assumed that the trap stage can not begin until the sixty-four-bit operation is complete, so the trap stage can not be entered until after the second, helper, cycle of the addition 301a; a stall stage 307 of the pipeline is utilized to ensure this. Stall stage 307 may take the form of a recycling of the operation within a stage of the pipeline, assuming that that stage has adequate storage in a recycle buffer for the longer operands, intermediate or final results.

Therefore, if no stall is injected into the thirty-two-bit subcommand, the sixty-four-bit subcommand will reach its trap cycle 305 after the thirty-two-bit subcommand reaches its trap cycle 306.

In a processor according to the present invention, instructions are received from the instruction cache 126 into an instruction aligner 400 (Figure 4) and instruction buffer 402, and processed by an instruction decoder and dispatcher 404. A helper subcommand inserter 406, which may be a part of the instruction decoder and dispatcher 404, inserts helper subcommands as required for proper execution of any sixty-four-bit or other subcommands that require additional time in execution.

Consider the case where a sixty-four-bit addition is executed in a first pipeline 410 of the processor, the first pipeline being based upon thirty-two-bit data paths. A fetch stage 412 of the pipeline therefore fetches the low halves of the operands in a cycle 500 (figure 5), the fetch stage 412 fetches the high halves of the operands in a following cycle 501, while the operate stage 414 executes the addition on the low halves of the operands 502. In the next cycle, while the helper subcommand executes 504 in the operate stage 414, the results of the low half of the operands is held 506 in a recycle buffer 416 of the operate stage, or in a stall stage of the pipeline. In the next cycle 510, any trap conditions are resolved, and in the next cycles 512 and 514 the results of the operation are stored in the register file 420 by storage stage 418 of the pipeline.

When a sixty-four-bit subcommand is processed that requires one cycle of operation before the trap stage beyond the timing of a thirty-two-bit subcommand from the same VLIW instruction word, the helper subcommand inserter, 406, also inserts a NOP, or no-operation, helper subcommand into the instruction

stream flow to the second pipeline 425 in which that thirty-two-bit subcommand executes, and marks that subcommand as having a stall. Each NOP helper subcommand, also known as a helper stall subcommand, is inserted after the subcommand dispatched to a pipeline, and causes data at the stage prior to the trap stage of the associated pipeline to remain unchanged, or be recycled, for one cycle. This causes data at the stage prior to the trap stage to remain unchanged until the last of any sequence of helper stall subcommands completes. This NOP helper subcommand is not injected if the first pipeline 410 receives a thirty-two-bit subcommand having similar timing to that intended for the second pipeline 425. That thirty-two-bit subcommand therefore executes with a fetch in the first cycle 530 and a NOP in the second cycle 532 at the fetch stage. In the second cycle the operate stage executes the operation 534, and in the third cycle the operate stage does a NOP 536. Since the subcommand is marked as having a stall, the results of the operation are held 538, such that they enter the trap cycle 540, or stage 430, of the second pipeline 425, simultaneously with the results of the sixty-four-bit subcommand executing in the first pipeline 410 reaching its trap stage 422. The results of the long, sixty-four-bit, operation and the shorter, thirty-two-bit operation, are therefore synchronized at the trap stages.

In an alternative embodiment, when a sixty-four-bit subcommand is processed that requires one cycle of operation before the trap stage beyond the timing of a thirty-two-bit subcommand from the same VLIW instruction word, the helper subcommand inserter, 406, also inserts a NOP, or no-operation, helper subcommand into the instruction stream flow to the second pipeline 425 in which that thirty-two-bit subcommand executes. Each NOP helper subcommand, also known as a

helper stall subcommand, is dispatched before of the associated thirty-two-bit instruction, as shown in Figure 6. This NOP helper subcommand is not injected if the first pipeline 410 receives a thirty-two-bit subcommand having similar timing to that intended for the second pipeline 425. Execution of the sixty-four-bit subcommand in this embodiment is as shown in Figure 5.

The thirty-two-bit subcommand therefore executes with a fetch in the second cycle 630 and a NOP in the first cycle 632 at the fetch stage. In the third cycle the operate stage executes the operation 534, and in the second cycle the operate stage does a NOP 536. Data associated with the subcommand then enters the trap cycle 640, or stage 430, of the second pipeline 425, simultaneously with the results of the sixty-four-bit subcommand executing in the first pipeline 410 reaching its trap stage 422 at 512. The results of the long, sixty-four-bit, operation and the shorter, thirty-two-bit operation, are therefore synchronized at the trap stages.

There are subcommands where multiple NOP helper subcommands must be entered into the pipeline to ensure that the results of a sixty-four-bit subcommand are synchronized with the results of a thirty-two-bit subcommand at the trap stage. In the preferred embodiment, sixty-four-bit multiplication operations are performed in a thirty-two-bit array multiplier. Multiplying a pair of sixty-four-bit operands in a thirty-two-bit multiplier, generating a one-hundred-twenty-eight-bit result, requires four passes through the multiplier. The VLIW instruction word is therefore decoded to insert three helper subcommands into the instruction stream for a pipeline receiving a sixty-four-bit multiply subcommand. The VLIW instruction word having such a multiply subcommand is decoded to inject three NOP helper subcommands after

each simultaneously executed thirty-two-bit addition or subtraction operation, instead of the one required when a sixty-four-bit addition is executed. Similarly, any simultaneously executing sixty-four-bit addition receives two NOP helper subcommands such that its results will be held in the recycle buffers of the execute stage and enter the trap stage simultaneously with the results of the sixty-four-bit multiply subcommand.

While the invention has been particularly shown and described with reference to a preferred embodiment thereof, it will be understood by those skilled in the art that various other changes in the form and details may be made without departing from the spirit and scope of the invention. In particular, it is expected that the number of pipeline stages may vary from those discussed, and that operations other than addition and subtraction may be executed. It is also expected that the present invention is applicable to machines that process data in words of other than the thirty-two- and sixty-four-bit lengths operating on thirty-two-bit hardware herein disclosed, it being applicable to machines having multiple pipelines where a pipeline may execute upon a longer data word than the width of the hardware.

It is also expected that the no-operation helper subcommand herein described as following a thirty-two-bit subcommand may instead be injected ahead of the thirty-two-bit subcommand, such that the results of the thirty-two-bit subcommand reach the trap stage simultaneously with the results of a sixty-four-bit subcommand.

CLAIMS

What is claimed is

1. A VLIW processor comprising:

a first pipeline for executing subcommands, and having at least one execution stage and a trap stage;

a second pipeline for executing subcommands, and having at least one execution stage and a trap stage, the second pipeline capable of operating in parallel with the first pipeline, the first pipeline capable of operation upon operands of a first word length and a second word length, where the second word length is greater than the first word length, where the first word length is the same as a data path width of the first pipeline, and where execution of operations on the operands of the second word length requires multiple cycles in at least one execution stage of the first pipeline;

an instruction decoder for decoding subcommands of a sequence of VLIW instructions into pipeline subcommands, and for dispatching these to the first pipeline and the second pipeline, the instruction decoder injecting at least one helper subcommand into the first pipeline when a first subcommand of the VLIW instruction is a command for operation upon operands of the second word length and is dispatched to the first pipeline;

wherein the instruction decoder is capable of inserting no-operation helper subcommands into the second pipeline when necessary to ensure that information associated with the first subcommand enters a trap stage of the first pipeline simultaneously with information associated with a second subcommand of the same VLIW instruction and dispatched to the second pipeline arriving at a trap stage of the second pipeline,

and wherein injection of the no-operation helper subcommands is conditioned upon the first subcommand and the second subcommand.

2. The VLIW processor of claim 1, wherein the information associated with the first subcommand that enters the trap stage of the first pipeline is a result of an arithmetic operation performed on operands of the second word length, produced by execution of a first pipeline subcommand and a helper subcommand in the first pipeline, and wherein the information associated with the second subcommand that enters the trap stage of the second pipeline is a result of an arithmetic operation performed on operands of the first word length.

3. The VLIW processor of Claim 1, wherein the first word length is thirty-two bits and the second word length is sixty-four bits.

4. A method of synchronizing the arrival of data in a plurality of pipelines of a processor to trap stages of the plurality of pipelines, comprising the steps of:

decoding an instruction of the processor into at least one subcommand, each subcommand intended for execution on a pipeline of the processor, wherein at least some of the set of possible instructions of the processor are instructions that decode into a plurality of subcommands;

dispatching the at least one subcommand to pipelines of the processor;

determining a number of helper stall subcommands associated with each subcommand, the number of helper stall subcommands selected from the group consisting of the set of non-negative integers, for injection into each pipeline receiving a subcommand, the number of helper stall subcommands being determined such that

information associated with execution of each subcommand decoded from a given instruction will execute the trap stage of the pipeline to which that subcommand is dispatched simultaneously with information associated with execution of any other subcommand decoded from the same instruction;

injecting the determined number of helper stall subcommands into each pipeline to which a subcommand of the instruction is dispatched.

5. The method of Claim 4, wherein the step of determining a number of helper stall subcommands is performed by decoding a bit pattern selected from the group consisting of the instruction and the set of subcommands decoded from the instruction.

6. The method of Claim 5, wherein the pipelines are capable of executing subcommands on data of width N bits and on width two times N bits, for some positive integer N, and where for a given operation execution to a trap stage on data of width two times N bits takes M more cycles of a clock than execution to a trap stage on data of width N bits, where M is a positive integer greater than zero.

7. The method of Claim 6, wherein a first subcommand executes on data of width N bits, a second subcommand executes on data of width two times N bits, and wherein the number of helper subcommands injected into the pipeline to which the first subcommand is dispatched is M.

8. The method of Claim 5, wherein the helper stall subcommands are injected into at least one of the pipelines before the subcommand dispatched to that pipeline for execution.

9. The method of Claim 5, wherein the helper stall subcommands are injected into at least one of the

pipelines after the subcommand dispatched to that pipeline for execution.

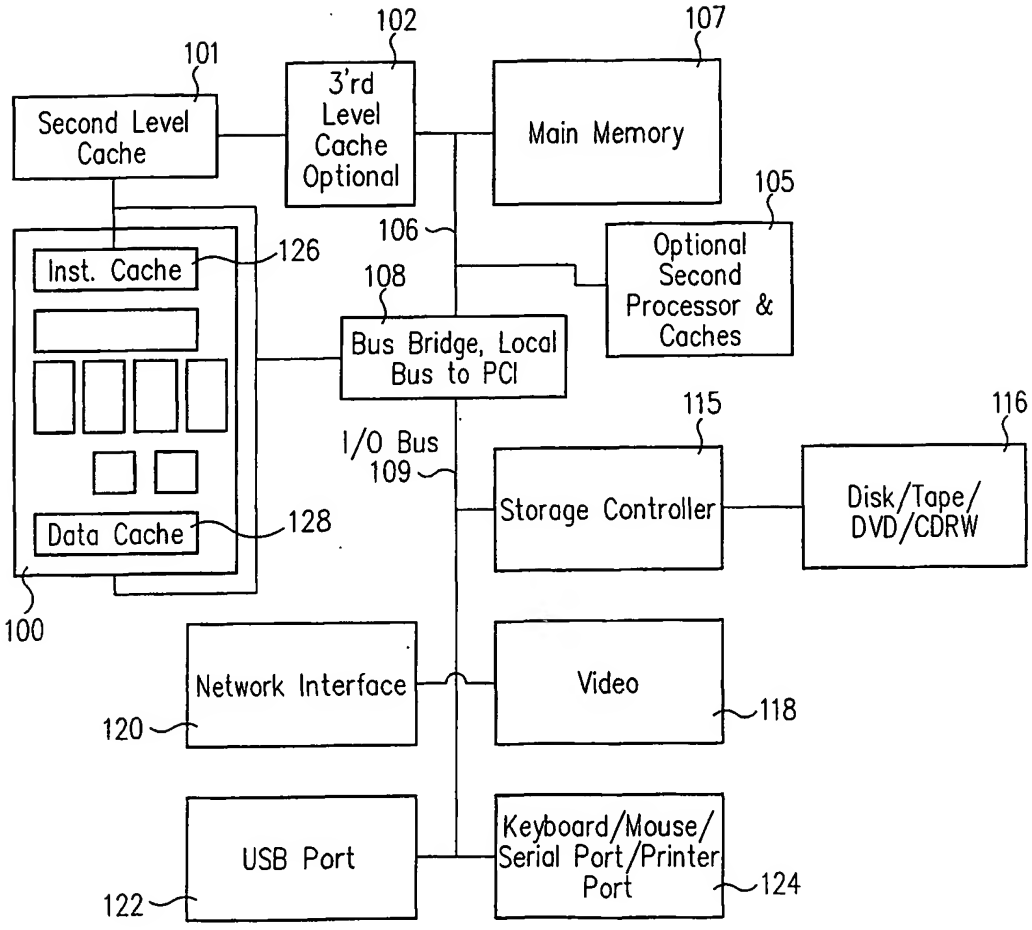


FIG. 1
(PRIOR ART)

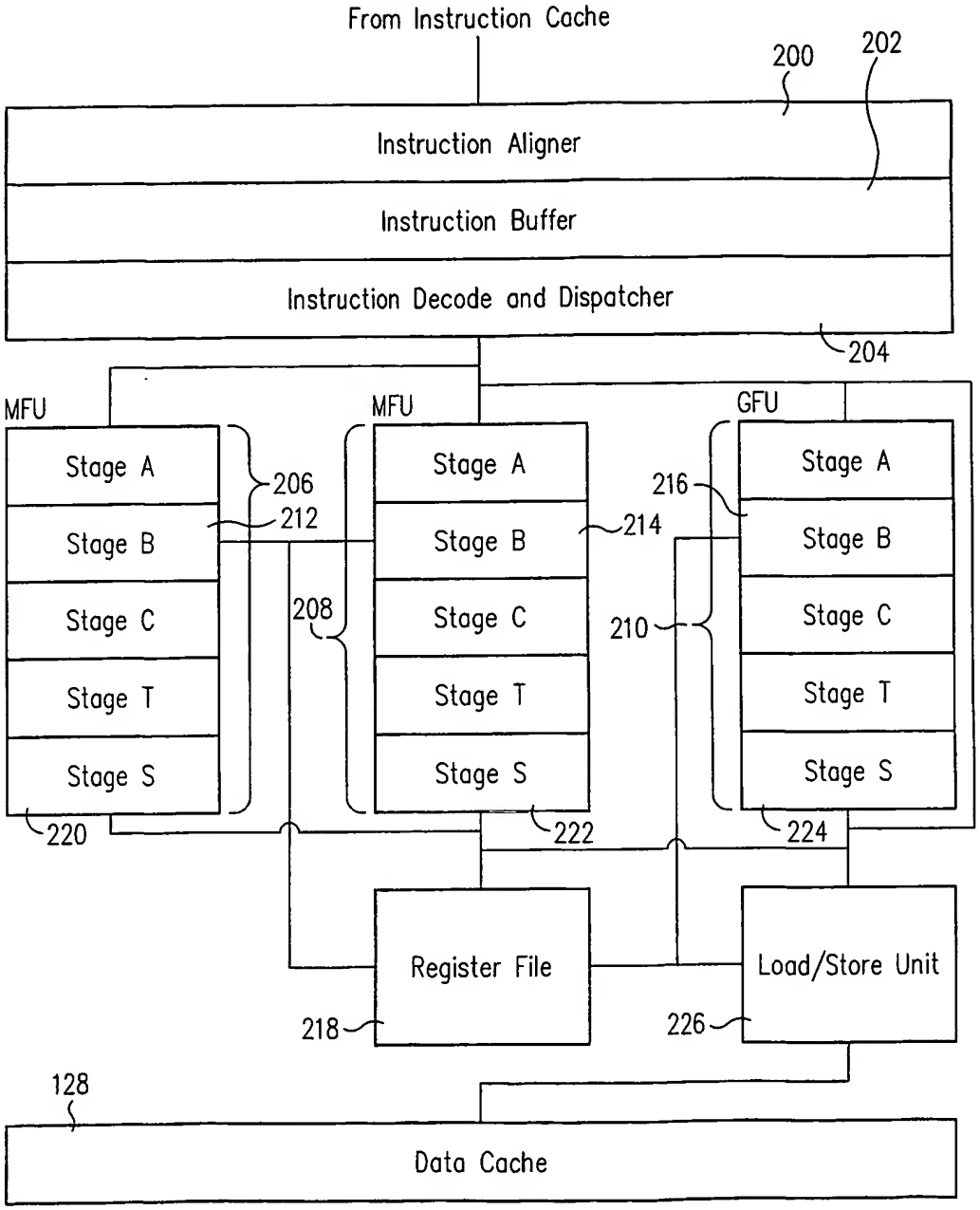


FIG. 2
(PRIOR ART)

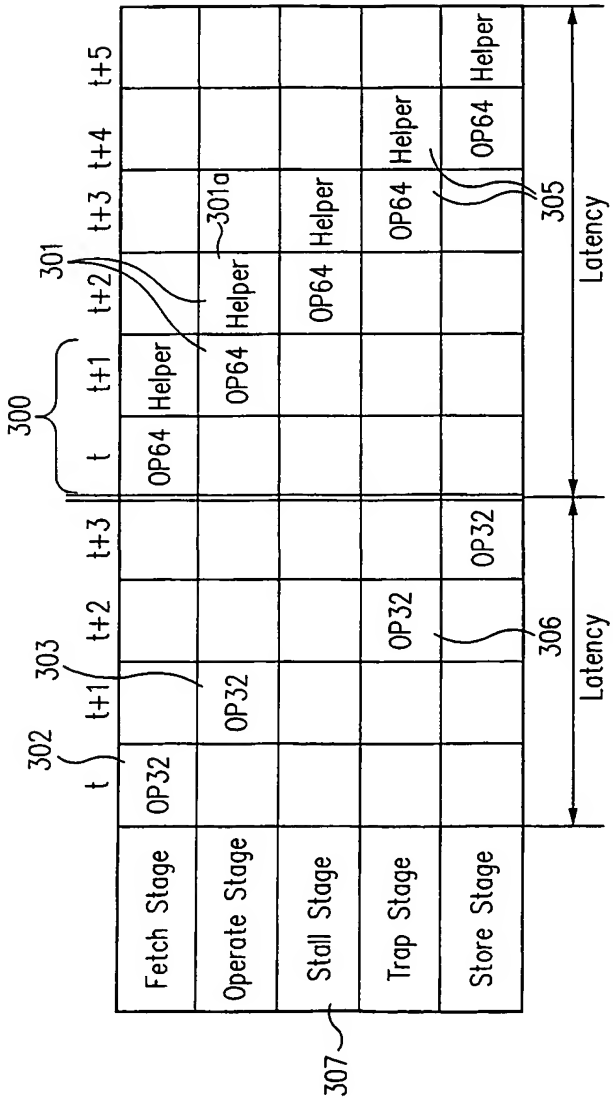


FIG. 3
(PRIOR ART)

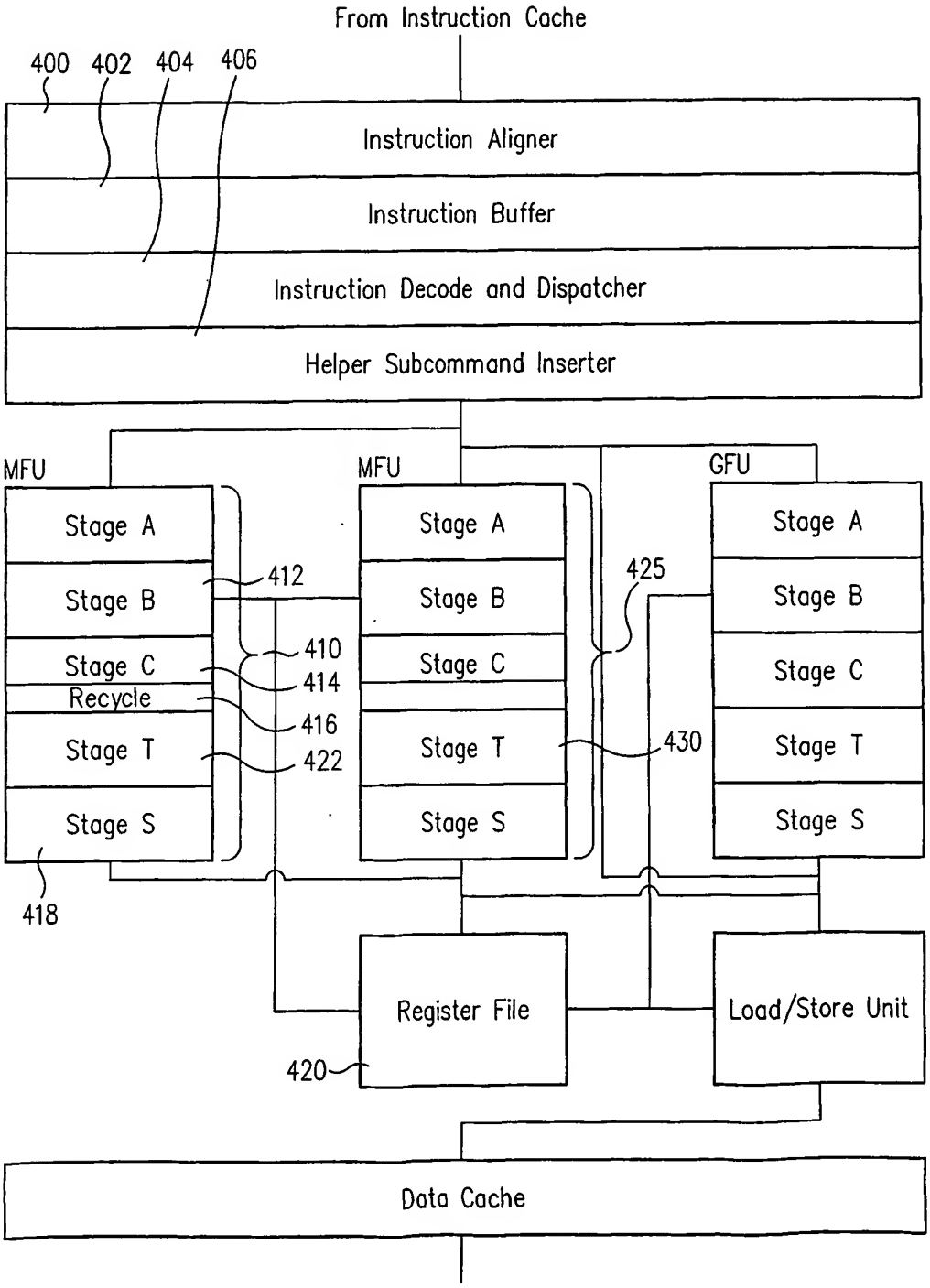

From Instruction Cache
400
402
404
406
Instruction Aligner
Instruction Buffer
Instruction Decode and Dispatcher
Helper Subcommand Inserter
MFU
Stage A
Stage B
Stage C
Recycle
Stage T
Stage S
MFU
Stage A
Stage B
Stage C
Stage T
Stage S
GFU
Stage A
Stage B
Stage C
Stage T
Stage S
412
410
414
416
422
425
430
418
Register File
420
Load/Store Unit
Data Cache

FIG. 4

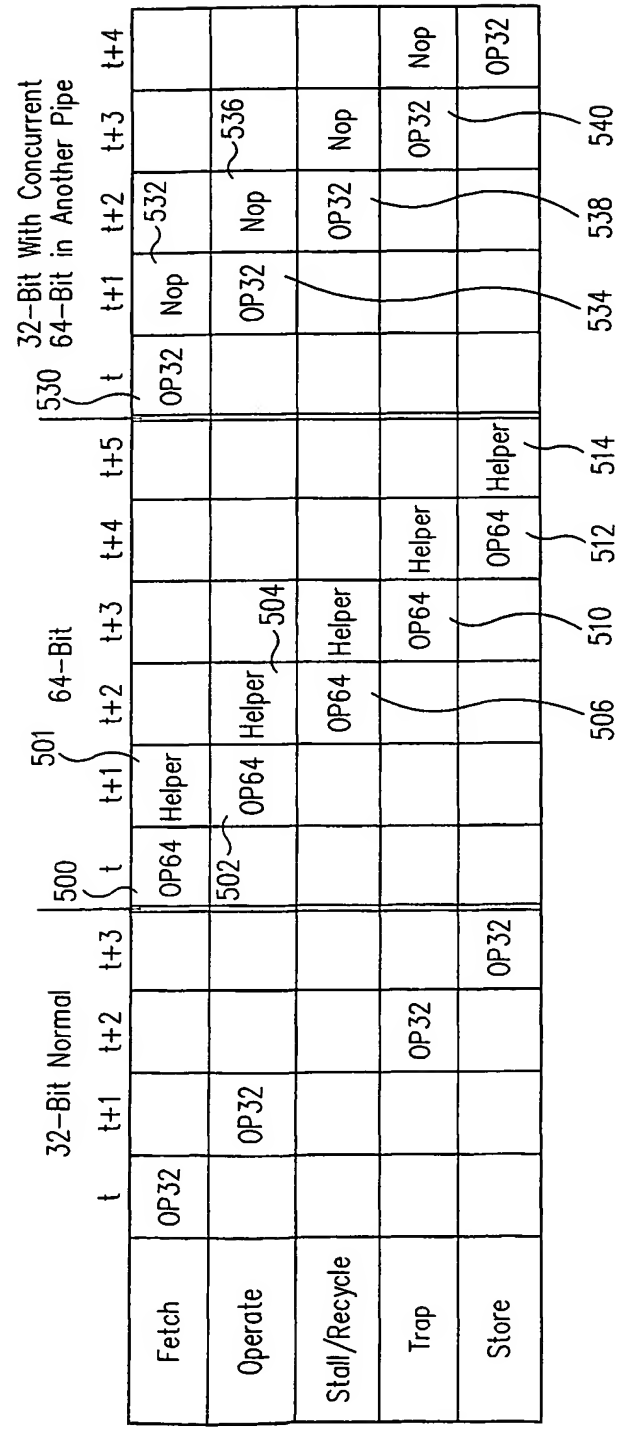

32-Bit Normal
32-Bit With Concurrent 64-Bit in Another Pipe
64-Bit
t
t+1
t+2
t+3
t+4
t+5
Fetch
Operate
Stall/Recycle
Trap
Store
OP32
OP64
Helper
Nop
500
501
502
504
506
510
512
514
530
532
534
536
538
540

FIG. 5

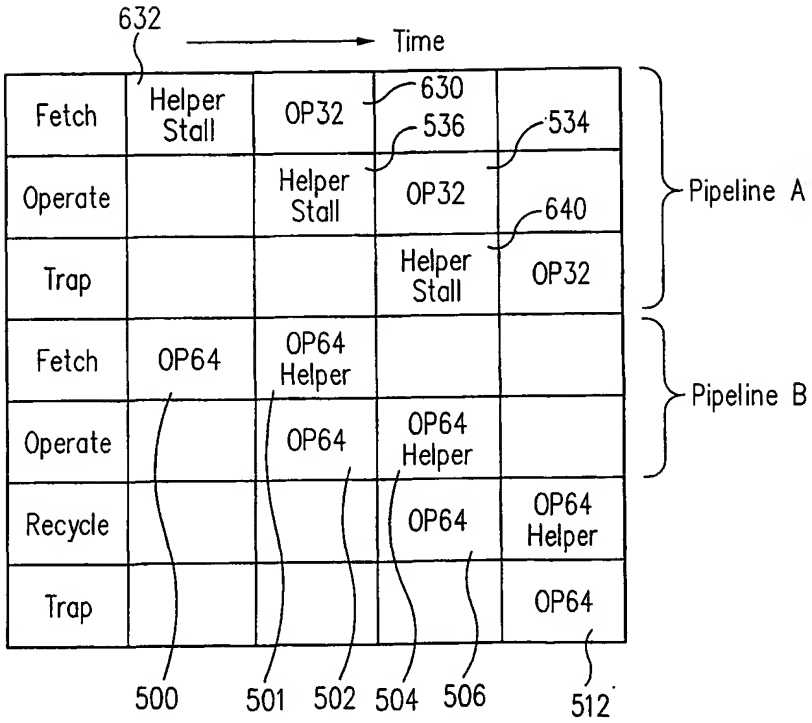

632
Time
Fetch
Helper Stall
OP32
630
536
534
Operate
Helper Stall
OP32
640
Trap
Helper Stall
OP32
Pipeline A
Fetch
OP64
OP64 Helper
Operate
OP64
OP64 Helper
Recycle
OP64
OP64 Helper
Trap
OP64
Pipeline B
500
501
502
504
506
512

FIG. 6

(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
13 December 2001 (13.12.2001)

PCT

(10) International Publication Number
WO 01/95101 A3

(51) International Patent Classification[7]: G06F 9/38, 9/302

(21) International Application Number: PCT/US01/10839

(22) International Filing Date: 30 May 2001 (30.05.2001)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09/586,190 2 June 2000 (02.06.2000) US

(71) Applicant: SUN MICROSYSTEMS, INC. [US/US]; 901 San Antonio Road, Palo Alto, CA 94303 (US).

(72) Inventors: TREMBLAY, Marc; 140 Hanna Way, Menlo Park, CA 94025 (US). YELURI, Sharada; 1720 Fumia Drive, San Jose, CA 95131 (US). CHAN, Jeffrey, Meng, Wah; 1984 Latham Street #10, Mountain View, CA 94040 (US).

(74) Agent: MCKAY, Philip, J.; Gunnison, McKay & Hodgson, L.L.P., Suite 220, 1900 Garden Road, Monterey, CA 93940 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW.

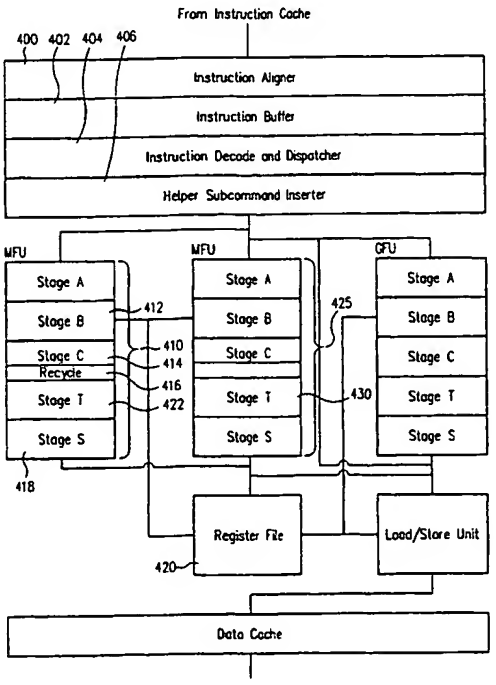
(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii)) for the following designations AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH,*

[Continued on next page]

(54) Title: SYNCHRONIZING PARTIALLY PIPELINED INSTRUCTIONS IN VLIW PROCESSORS



(57) Abstract: AVLIW processor has multiple pipelines (410, 425) for execution of subcommands of VLIW instructions in parallel. Each pipeline has at least one execution stage (412, 414) and a trap stage (422, 430). At least one can operate on operands of a first and a second word length, the second word length longer than the first, the first word length is the same as a data path width of the pipeline (410, 425). Execution of operations on the operands of the second word length requires multiple cycles in at least one execution stage (412, 414) of the pipeline. An instruction decoder (404) decodes subcommands of a sequence of VLIW instructions into pipeline subcommands, and dispatches these to the first and second pipelines (410, 425), the instruction decoder (404) injects at least one helper subcommand into the first pipeline (410) when a first subcommand of the VLIW instruction operates on operands of the second word length. The instruction decoder also inserts no-operation helper subcommands into the second pipeline (425) when necessary to ensure that information associated with the first subcommand enters a trap stage (422) of the first pipeline (410) synchronously with information associated with a second subcommand of the same VLIW instruction and dispatched to the second pipeline (425) reaching a trap stage (430) of the second pipeline (425). These no-operation helper subcommands maintain synchronous arrival of information at the trap stages (422, 425) even if the first subcommand operates on operands of the second word length and the second subcommands operates on operands of the first word length.

WO 01/95101 A3

CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG)

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii)) for the following designations AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG)*

Published:

— *with international search report*

(88) Date of publication of the international search report:
21 March 2002

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

INTERNATIONAL SEARCH REPORT

International Application No
PCT/US 01/10839

A. CLASSIFICATION OF SUBJECT MATTER
IPC 7 G06F9/38 G06F9/302

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 7 G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	EP 0 730 223 A (MATSUSHITA ELECTRIC IND CO LTD) 4 September 1996 (1996-09-04) the whole document	1-9
A	EP 0 653 703 A (SUN MICROSYSTEMS INC) 17 May 1995 (1995-05-17) column 2, line 27 -column 3, line 57	1-9
P,A	WO 00 33183 A (SUN MICROSYSTEMS INC) 8 June 2000 (2000-06-08) the whole document	1-9
A	EP 0 649 085 A (CYRIX CORP) 19 April 1995 (1995-04-19)	

☐ Further documents are listed in the continuation of box C

☒ Patent family members are listed in annex.

* Special categories of cited documents

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
7 December 2001	18/12/2001

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax (+31-70) 340-3016	Moraiti, M

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International Application No
PCT/US 01/10839

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0730223 A	04-09-1996	EP 0730223 A1	04-09-1996
		JP 2869376 B2	10-03-1999
		JP 8305566 A	22-11-1996
		KR 180580 B1	15-05-1999
		US 5822561 A	13-10-1998
EP 0653703 A	17-05-1995	US 6128721 A	03-10-2000
		DE 69418146 D1	02-06-1999
		DE 69418146 T2	25-11-1999
		EP 0653703 A1	17-05-1995
		JP 7191846 A	28-07-1995
WO 0033183 A	08-06-2000	US 6279100 B1	21-08-2001
		WO 0033183 A1	08-06-2000
EP 0649085 A	19-04-1995	DE 69408769 D1	09-04-1998
		DE 69408769 T2	09-07-1998
		EP 0649085 A1	19-04-1995
		JP 7152559 A	16-06-1995
		US 5630149 A	13-05-1997
		US 5784589 A	21-07-1998
		US 6138230 A	24-10-2000
		US 6073231 A	06-06-2000